resource adapter

WEST

Freeform Search

Database:
- US Patents Full-Text Database
- US Pre-Grant Publication Full-Text Database
- JPO Abstracts Database
- EPO Abstracts Database
- Derwent World Patents Index
- IBM Technical Disclosure Bulletins

Term: L6 and (java$)

Display: 10 **Documents in Display Format:** KWIC **Starting with Number** 1

Generate: ○ Hit List ◉ Hit Count ○ Side by Side ○ Image

Search | Clear | Help | Logout | Interrupt

Main Menu | Show S Numbers | Edit S Numbers | Preferences | Cases

Search History

DATE: Monday, March 10, 2003 Printable Copy Create Case

Set Name side by side	Query	Hit Count	Set Name result set
DB=USPT; PLUR=YES; OP=ADJ			
L7	L6 and (java$)	3	L7
L6	(resource adj adapter)	43	L6
L5	L1 and (resource adj adapter)	0	L5
L4	L3 and java	0	L4
L3	(connector adj contract$)	36	L3
L2	l1 and (connector adj contract$)	0	L2
L1	(java adj class$)	578	L1

END OF SEARCH HISTORY

L7: Entry 1 of 3 File: USPT Dec 24, 2002

DOCUMENT-IDENTIFIER: US 6499036 B1
TITLE: Method and apparatus for data item movement between disparate sources and hierarchical, object-oriented representation

Brief Summary Text (6):
Modern business enterprises face another challenge in matching their computer applications to their data assets. The majority of data assets are maintained using traditional data management systems and techniques. Some reasons for this are the huge investment made in building and maintaining the existing assets, the proven reliability of the existing systems, and the cost of migrating to more modem systems. Object oriented databases are on the horizon, moving from the laboratory to the field, but have yet to make serious inroads to widespread commercial use. But while structured data management systems have lagged in adopting an object-oriented design paradigm, commercial application software designs have fully embraced it. Use of object-oriented programming languages, such as C++ and Java proliferates. Thus, modem business organizations face a schism between the design paradigm underlying their application software and the design paradigm managing the data on which the software is to operate.

Detailed Description Text (18):
A programmer uses development software 262 to create an application program that will run on a client computer 120. Development software 262 may include source code editors, integrated development environments, language compilers, linkers, debuggers, and other tools used by programmers to develop software. In the preferred embodiment these tools support development in an object oriented language such as C++ or JAVA. Such tools are well known and understood in the art.

Detailed Description Text (36):
FIG. 4 depicts a functional architecture employing the present invention. The described embodiment utilizes a layered architecture to mediate the end-to-end movement of data items. At one end is user application code 490 that manipulates a transient copy of a data item. At the other end are stored data 432 which contain persistent copies of data items. Intervening between the ends are a client transactor layer, a client-server (CS) communication layer, a server processor layer, a resource adapter layer, a server-source (SS) communication layer, and a data access layer. Client transactor layer comprises executing Foundation for Distributed Object (FDO) program code 450 that functions to interface user application code 490 to the request and response data formats and protocols required by the DOM server software 210. CS communication layer comprises CS communication channel 460 that functions to transfer request and response related data bi-directionally between the client transactor layer and the server processor layer. Server processor layer comprises DOM core program code 410 that functions to receive, process, and respond to transaction requests originating from user application program code 490. The server processor layer processes the requests it receives by initiating SS transactions with data sources 430. Resource adapter layer comprises resource adapters 412 that function to receive SS transaction requests from the DOM core and to conduct the requested transaction activity with an associated data source on behalf of, and under the direction of, the DOM core. SS communication layer comprises communication channels 471 that function to transfer SS transaction request and response related data bi-directionally between the resource adapter layer and the data accessor layer. Data accessor layer comprises data accessor programs 431 that function to maintain and manage the persistent data items contained in stored data 432.

Detailed Description Text (43):

DOM server computer 110 executes DOM server software 210 comprising the program code of process activation daemon (PAD) 416, protocol-specific fa.cedilla.ade 414, resource adapters 412, and DOM core 410. DOM server computer 110 connects to data sources 430 via communication channels 470. DOM server software 210 executes using execution processes 480-484.

Detailed Description Text (53):
For each SS transaction request that needs to be directed to a data source, the DOM core 410 connects to program logic in one of resource adapters 412 to initiate the transaction with the associated data source. As to transactions between the DOM server software 210 and the data sources 430, the DOM server software acts as the client, and the data sources each act as a server. For example, the DOM core 410 connects to resource adapter 412a to begin an exchange with data source 430a. Resource adapter 412a communicates with DOM core program code 410 in a format commonly used by all resource adapters 412. Resource adapter 412a accepts requests in the common DOM format and performs the requested transaction by communicating with data source 430a in accordance with requirements imposed by communication channel 471a and data access software 431a.

Detailed Description Text (54):
Communication channel 471a connects resource adapter 412a to data access software 431a. Communication channel 471a is a bi-directional data communication channel. Communication channel 471a may comprise a physical communication medium interposed between support circuitry and software at each end. The channel 471a may provide for communication between programs executing on the same computer, or on computers located thousands of miles apart. Examples of communication channels that may be employed in the presently described embodiment include TCP/IP network connection and SDLC network connections. These and other inter-program communication and interaction facilities are well known and understood in the art.

Detailed Description Text (56):
Resource adapter 412a and communication channel 471a together operate as described above to give the DOM core access to data source 430a. Resource adapter 412b and communication channel 471b operate similarly to give the DOM core access to data source 430b. Resource adapter 412c and communication channel 471c operate similarly to give the DOM core access to data source 430c. In the presently described embodiment there are only practical limits, such as memory and CPU capacity, restricting the number of data access pipelines 470 associated with an executing DOM server 210. The ability of the DOM server 210 to simultaneously interact with multiple and disparate data sources represents a further advantage of the present invention.

Detailed Description Text (57):
The DOM server software 210 depicted in FIG. 4 also includes process activation daemon (PAD) 416. PAD 416 program logic loads and executes on server machine 110 to initiate operation of the DOM server software 210. PAD 416 reads configuration information from configuration database 115. Based on the information obtained from configuration database 115, PAD 416 starts multiple processes 482-484. Each process is viewed by the operating system software of server computer 110 as an independently manageable and dispatchable unit of work. PAD 416 starts processes 484, and 482a-c, to execute DOM core 314, and resource adapter 412a-c program logic, respectively. Starting each process includes loading the program code that directs the execution of the process. For example, starting process 484 includes loading DOM core program code 410 from mass storage into main memory.

Detailed Description Text (59):
PAD 416 starts resource adapter processes 412a-c, after starting DOM core process 484 is shown. In the presently described embodiment one process 484 executing DOM core program logic 410. All inbound communication channels 462 from client machines, and all outbound communication channels 471a-c to data sources, connect to the single executing copy of DOM core program logic 410. In other embodiments, multiple processes running DOM core program logic may be started on a single server machine 110. Inbound requests from client machines and outbound requests to data sources may then be distributed among the multiple executing DOM core processes to manage or enhance performance. One skilled in the art recognizes that these and other

variations can be made without departing from the scope and spirit of the invention.

Detailed Description Text (62):
Steps 520 through 540 are performed using the GUI configuration utility described earlier. Steps 520 through 540 populate the DOM configuration database 115 which stores information used to direct the operation of the DOM server software during its execution. Step 520 records configuration information for the DOM server's Process Activation Daemon (PAD). Step 520 records its output in PAD Configuration file 572, which is contained in configuration database 115. PAD Configuration file 572 contains three sections. An ENVIRONMENT section contains information about the identity and location of computing resources on the server host. Such resources may include, for example, file directories or communication ports. RESOURCE ADAPTER and DOM sections contain information the PAD needs to start, control, and monitor resource adapter and DOM core processes. Such information may include, for example, a name that identifies a computer to run the process; an indicator whether restart should be attempted for a failed process; how many times, how often, and how long to wait for restart attempts; information the operating system should provide to the process if specifically requested; the command string to submit to the operating system to start the process; and the command string to submit to the operating system to restart the process. The RESOURCE ADAPTER and DOM sections may additionally include information to facilitate concurrent execution of multiple resource adapters or DOM cores on the server machine. Such information may include, for example, the number of DOM or resource adapter occurrences the PAD is to run.

Detailed Description Text (63):
Step 530 records configuration information used by executing resource adapters. DOM configuration "resources" as defined in step 530 refer to data sources. Each resource definition corresponds to a run-time resource adapter process and the SS transactions that the particular resource adapter can perform. The output of resource definition step 530 corresponding to a particular resource adapter goes to a particular resource adapter configuration file 574. Resource adapter configuration file 574 is contained in configuration database 115. Configuration database 115 in the presently described embodiment contains one resource adapter configuration file for each resource adapter specified in PAD configuration file 572.

Detailed Description Text (64):
A resource definition contains information concerning the resource adapter. For example, such information may include the type of data source to which the resource adapter connects; location, authorization, and interface information used to establish secure and reliable communication with the data source; and process management information about the resource adapter such as the amount of time to wait for initialization, restart parameters, environment information such as file directory names, a shell command for starting the resource adapter, and whether multiple, concurrent copies of the resource adapter may execute.

Detailed Description Text (65):
A resource definition also contains information concerning each particular SS transaction that the related resource adapter can accommodate.

Detailed Description Text (66):
For example, information about a transaction may include an identifier for the transaction type, parameters to control transaction execution, the format of the transaction request, the format of the transaction reply, the correspondence between transaction data items and DOM server data container data items, and procedural logic needed to conduct the transaction. Because different resource adapters can connect to different types of data sources, and because different data sources may each have their own particular defined interfaces, the specific information required to engage in a transaction may vary from data source type to data source type. In a preferred embodiment, each transaction definition corresponds to what the data source considers to be a single, atomic transaction. i.e., one request-process-reply sequence. The atomic transaction may also include required handshaking such as session set-up and tear-down.

Detailed Description Text (69):

Step 550 creates a client application program. The output of client program creation step 550 is an executable program 242 in mass storage that can be loaded and executed in a client computer. In a preferred embodiment, the client program is written by a computer programmer using an object-oriented computer language such as C++ or Java. The programmer includes source code in the program to send a request to a DOM server. The request is formulated to invoke a method defined in step 540. The source code is compiled and linked to form executable program 242. Application program 242 will control the operation of a client computer at execution time. After step 550, the implementation process is complete.

Detailed Description Text (77):
Method display area 650 graphically depicts the configuration information associated with any DOM configuration file in a configuration database. This information defines and describes the client-server (CS) transactions available to a client application program through a DOM server. Resources display area 660 graphically depicts the configuration information associated with any resource adapter configuration file in a configuration database.

Detailed Description Text (80):
FIGS. 7 through 13 illustrate GUI configuration for resource adapters and SS transactions. FIGS. 7 through 10 illustrate GUI configuration for a CICS-EPI resource and three associated SS transactions. FIGS. 11 through 13 illustrate GUI configuration for an RDBMS resource and one associated SS transaction. The configuration activity represented by FIGS. 7 through 13 corresponds to the activity represented by step 530 of FIG. 5.

Detailed Description Text (81):
FIG. 7 illustrates the configuration of a transaction monitoring software system (CICS-EPI) used as a data source. The "EPI (CicsEpi)" element 791 of the visible contents 662 from the resource display area (660 of FIG. 6) of the DOM Configurator main screen (600 in FIG. 6) represents the configuration information for a CICS-EPI data source 130a. After adding element 791 to the resource display area, the user of the DOM Configurator displays dialog box 700 to view and modify the configuration information represented by element 791. Dialog box 700 comprises title bar 701, command buttons 702,703, execution environment information area 710, resource adapter management area 720, and resource adapter-specific information area 730. Title bar 701 displays the name of the type of resource adapter being configured. Command buttons 702 and 703 may be individually selected by the user using a keyboard or pointing device. Command button 702, when selected, invokes processing to terminate the display of the dialog box. Any changes made by the user to information displayed in the dialog box are saved during such termination processing. Command button 703, when selected, also invokes processing to terminate the display of the dialog box, but without saving any changes made by. the user.

Detailed Description Text (82):
Dialog box area 710 displays information about the execution environment of the target data source. Such information may include, for example, whether the data source resides on the same or different computer as the resource adapter software, or information about the coding methods used to represent data on the computer where the target data source executes.

Detailed Description Text (83):
Dialog box area 720 displays information principally used by the PAD software to manage any process executing a resource adapter using the instant configuration. The "Resource Type" field indicates the kind of data source to which the resource adapter connects. The "Resource Name" field indicates a name to be used by other software components to identify the resource adapter. The "Config File Name" field indicates the name of a resource adapter file in the configuration database where the instant configuration information is stored. The "Host Name" field indicates the name by which the data source computer can be contacted by software components attached to a common communication system. The "Instances" field indicates the number of concurrent resource adapters that may execute using the instant configuration information. The "Command" field indicates a command that the PAD may send to the operating system to initiate the resource adapter process. The "Restart Command" field indicates a command that the PAD may send to the operating system to

attempt a restart of the resource adapter process after an earlier failure. The "Retries(number, sec)" fields indicate the number of times the PAD issues the restart command if the previous start or restart command fails within the specified number of seconds. The "Wait" field indicates the amount of time that PAD should wait after issuing a start or restart command to determine whether the command was successful. The "Restart" field indicates whether PAD should issue a restart command after detection of an earlier start or restart failure. The "Environment" field indicates information that the operating system should make available to the executing resource adapter software should it make a request for such information using operating system services.

Detailed Description Text (84):
The information displayed in area 720 of resource dialog box 700 is common to all types of resource adapters. Particular types of resource adapters may require that specific additional information be included in their configuration files. Display area 730 of resource dialog box 700 accommodates such information. For example, the "CICS Region" field indicates the name of a particular execution copy of CICS-EPI software running on the target data source computer.

Detailed Description Text (85):
FIGS. 8 through 10 illustrate the configuration of SS transactions between a DOM server and a CICS-EPI data source. FIG. 8 illustrates the configuration of an SS transaction for a CICS-EPI data source. When element 791 for the EPI data source is added to the resource display area at the specific request of the user, "Transactions" element 891 is automatically available for display. "Transactions" element 891 serves as the anchor point for a list of server-source (SS) transactions that the EPI resource adapter can process. The user of the DOM Configurator specifically adds the "GetCustomer" display element 892 to the transactions list anchored by element 891. When GetCustomer 892 is added to the transactions list, "EPI Script" element is automatically anchored off of GetCustomer element 892, because configuration information represented by the "EPI Script" element 893 is necessary to conduct an SS transaction with a CICS-EPI-type data source.

Detailed Description Text (86):
Dialog box 810 illustrates the configuration information immediately represented by GetCustomer element 892. A "Name" field indicates a transaction name by which the CS transaction is known within the configuration database. The transaction name must be unique within the resource adapter configuration to which it belongs.

Detailed Description Text (90):
FIG. 11 illustrates the configuration of a relational database management software system (RDBMS) used as a data source. The configuration of the RDBMS data source parallels the configuration of the EPI data source illustrated and described in relation to FIG. 7. The "Informix (RDBMS)" element 1191 of the visible contents 664 from the resource display area (660 of FIG. 6) of the DOM Configurator main screen (600 in FIG. 6) represents the configuration information for an RDBMS data source 130e. Dialog box 1100 comprises execution environment information area 1110, resource adapter management area 1120, and resource adapter-specific information area 1130.

Detailed Description Text (91):
Dialog box area 1110 displays information about the execution environment of the target data source. The types of information contained in display area 1110 are the same as for display area 710 described earlier in reference to FIG. 7. Dialog box area 1120 displays information principally used by the PAD software to manage any process executing a resource adapter using the instant configuration. The types of information contained in display area 1120 are the same as for display area 720 described earlier in reference to FIG. 7. Display area 1130 contains no information, as no additional information beyond that accommodated in display areas 1110 and 1120 is needed to manage the execution of an RDBMS-type resource adapter FIG. 12 illustrates the configuration of a session definition as may be required by RDBMS software. When display element 1191 for the Informix RDBMS data source is added to the resource display area at the specific request of the user, "Sessions" element 1192 is automatically available for display. "Sessions" element 1192 serves as the anchor point for a list of logical sessions the resource adapter may establish with

the data source in order to interact with the data source. The user of the DOM Configurator specifically adds the "sys2_informix_session" display element 1291 to the sessions list anchored by element 1192. Each session element, such as 1291, represents configuration information for a logical connection to the RDBMS data source.

Detailed Description Text (92):
Dialog box 1200 displays the configuration information underlying session element 1291. The "Session Name" field indicates an identifier by which the instantly configured session may be known. The "Database Type" field indicates the name of a library file containing program code that can transform SS transaction requests and responses between a generalized RDBMS format and a format required by a particular RDBMS data source. The "Database Type" field is used because the design of this embodiment employs a generalized RDBMS resource adapter that is specialized to a particular RDBMS server by this association to a related library file. The "Server Name" field indicates the name by which the data source computer can be contacted by software components attached to a common communication system. The "Database Name" field indicates the name by which the RDBMS server identifies the particular database containing the data items targeted for access. The "Database Role" field indicates the function of connection; e.g., whether the connection is used to connect a client for data access requests, or whether the connection is used for performing data base administration activities. The "User ID" and "Password" fields indicate security parameters that will authenticate and authorize the resource adapter to utilize the services of the RDBMS data source. The "# of Connections" field indicates the number of logical connections the resource adapter should establish between itself and the data source using the instant configuration information. The "# of Output Rows" field indicates the default maximum number of table rows that should be included in the response to an SS transaction request originating from the resource adapter.

Detailed Description Text (93):
Notably, no sessions list is included in the configuration information for a CICS-EPI data source already described in relation to FIG. 7. This is because different data sources impose different interface requirements. In a preferred embodiment, the construction of the DOM configurator software is similar to that of the DOM server software wherein the resource adapters interface to the DOM core in a common fashion to provide a degree of modularity. In a preferred embodiment, resource adapter configuration maintenance code, including the program code to effectuate the dialog boxes related to the particular resource adapter, are modular in design and interface to "core" GUI-based DOM Configurator code using a common interface. Such a construction improves the upgradability of the DOM Configurator as new types of data sources become desired.

Detailed Description Text (95):
When display element 1191 for the Informix RDBMS data source is added to the resource display area at the specific request of the user, "Transactions" element 1193 is automatically available for display. "Transactions" element 1193 serves as the anchor point for a list of server-source (SS) transactions that the Informix RDBMS resource adapter can process. The user of the DOM Configurator specifically adds the "GetSecurity" display element 1391 to the transactions list anchored by graphical element 1193. When GetSecurity 1391 is added to the transactions list, "Parameters" 1392, "Field Map" 1393, and "SQL" 1395 elements are automatically anchored off of GetSecurity element 1391. Parameter 1392 and SQL 1393 configuration elements each directly possess underlying configuration information. Field map element 1393 serves as the anchor point for a list. Each entry in the field map list establishes the correspondence between a data item in a DOM data container object and a data item accessed using the RDBMS.

Detailed Description Text (96):
Dialog box 1310 illustrates the configuration information immediately represented by GetSecurity element 1391. A "Name" field indicates a transaction name by which the CS transaction is known within the configuration database. The transaction name must be unique within the resource adapter configuration to which it belongs.

Detailed Description Text (97):

Dialog box 1320 illustrates the configuration information represented by "Parameters" element 1392. A "Type" field indicates whether the database processing for the transaction is defined by structured query language (SQL) statements supplied to the data source by the resource adapter or by a predefined procedure stored on the data source computer. If a predefined procedure is to be used, the name of the procedure must be indicated in the "Proc Name" field. If SQL statements are to be supplied to the data source by the resource adapter, the SQL statements are specified using dialog box 1340. SQL statements specified using dialog box 1340 may include substitution variables such as the "$id" variable shown in dialog box 1340. SQL and variable substitution techniques are well known and understood in the art. A "Session Name" field in dialog box 1320 indicates the name of database session to utilize in conducting the transaction. Such a session must have already been configured in accordance with the discussion regarding FIG. 12. A "# of Output Rows" field indicates the maximum number of table rows that should be included in the response to the instant SS transaction request. This number overrides the default specified in the session configuration as discussed previously in reference to FIG. 12.

Detailed Description Text (113):
FIG. 16 illustrates a screen display for viewing configuration data in text format. The GUI-based Configurator provides the screen display depicted by window 1600 as an aid in troubleshooting and understanding. Window 1600 comprises "Resources" area 1610, "Resource Configuration" area 1620, and "DOM Configuration" area 1630. Resources area 1610 displays a list of the names of configured resources, i.e., data sources. The presently selected resource name appears as light text against a dark background. "Resource Configuration" area 1620 displays the configuration for the presently selected resource of area 1610 in text format. In the sample text depicted in display area 1620, "Method" refers to "SS transaction." The contents of display area 1620 corresponds to the contents of a Resource Adapter configuration file 576 as depicted in FIG. 5.

CLAIMS:

49. A data processing system for processing data maintained by disparate data sources, using object oriented programs, comprising: a client transactor layer for sending a primary transaction request, receiving a response to the primary transaction request, transmuting information from said response into programming objects, and for selecting the class of a primary application programming object from a set of candidate object classes based, at least in part, on information in said response, a client-server communication layer coupled to said client transactor layer for providing bidirectional communication; a server processor layer coupled to said client-server communication layer for receiving, processing, and responding to primary transaction requests, wherein said processing includes identifying and requesting server-source transactions useful in satisfying the primary transaction request; a resource adapter layer coupled to said server processor layer for receiving and communicating requests for server-source transactions; a server-source communication layer coupled to said resource adapter layer for providing bidirectional communication; a data accessor layer coupled to said server-source communication layer for supplying data from disparate sources in response to server-source requests communicated from said resource adapter layer; and wherein said responding of said server processor layer includes relating data items into said response according to a stored pattern that is independent of relationship patterns for said data items used by the data accessor layer.

WEST

Generate Collection

L7: Entry 2 of 3 File: USPT Nov 26, 2002

DOCUMENT-IDENTIFIER: US 6487208 B1
TITLE: On-line switch diagnostics

Detailed Description Text (122):
6. References [1] Systems Management Overview, ISO/IEC 10040 (X.701). [2] Management Framework for OSI, ISO/IEC 7498-4 (X.700). [3] Common Management Information Protocol (CMIP), ISO/IEC 9596. [4] The Common Management Information Services and Protocols for the Internet (CMOT and CMIP), RFC 1189. [5] A Simple Network Management Protocol (SNMP), RFC 1157. [6] Java Management API (JMAPI), http://java.sun.com/products/JavaManagement [7] DMI 2.0s Specification, http://www.dmtf.org/spec/spec.html [8] CORBA/IIOP 2.2, http://www.omg.org/library/specindx.html [9] CORBAservices, http://www.omg.org/library/specindx.html [10] The RS/6000 SP High-Performance Communication Network, http://www.rs6000.ibm.com/resource/technology/sp_sw1/spswp1.book.sub.-- 1.html. [11] UML Finite State Machine Diagrams, Robert C. Martin, Engineering Notebook Column, C++ Report, June, 1998.

Detailed Description Text (152):
Although these approaches may be used to implement the testing and diagnostic methods of the present invention, they have serious disadvantages: 1. Scripts are system administrators' home-made utilities, which mostly perform some automation of trivial testing (like pinging some nodes in order to test the nodes' connectivity). These scripts can test some very simple deterministic situations (such as a node that consistently does not respond to the ping), but are useless for more sophisticated testing. Also, system administrators usually do not have real knowledge about underlying hardware at a sufficient level in order to create more clever scripts. 2. User space programs designed to diagnose switch-related problems suffer from the fact that they cannot access privileged resources, such as adapters, switches, reported errors, etc. They are not able to create sufficient testing of the hardware or analyze the results of testing, since they have no integration or interfaces with switch management software that was created for such diagnostics purposes. 3. Diagnostics programs that are non-compatible with switch management software cannot run concurrently with applications. One extreme example of such diagnostics program is how advanced diagnostics implemented in most Windows NT boxes: user have to exit Windows NT and reboot the machine to DOS, and from there run the advanced diagnostics test suite.

Detailed Description Text (161):
The SPD object-oriented framework architecture provides a common framework that addresses all of these issues, providing a high degree of code reuse. The SPD framework has two aspects: the first is the framework itself, which provides all design and implementation patterns for the tests; and the second is helper utilities, which create a general interface to common resources, such as adapter use, protocol use, AIX resources use, etc.

Other Reference Publication (3):
Java Management API (JMAPI), http://java.sun.com/products/JavaManagement.

WEST

Generate Collection

L7: Entry 3 of 3 File: USPT Oct 15, 2002

DOCUMENT-IDENTIFIER: US 6466654 B1
TITLE: Personal virtual assistant with semantic tagging

Detailed Description Text (33):
The VA applications are analogous to web pages, determining the content that will be presented and controlling the interactions with the user. A VA application uses scripting languages such as VBScript, JavaScript, and Perl, so that developers can add significant functionality to a VA, such as performing mathematical calculations, processing text, and calling ActiveX and COM objects.

Detailed Description Text (116):
As shown in FIG. 12, the following telephony hardware is recommended to support a single ISDN PRI link, which can handle 23 simultaneous calls: 1 D/240PCI-T1 Adapter 210 1 D/320SC Voice Resource Adapter 212 1 DCB/320Conferencing Adapter 214 1 CTBUS to SCBus Connector 216 2 SCBus ribbon cables 218

Detailed Description Text (117):
As shown in FIG. 13, to support two ISDN PRI links, which can support 46 simultaneous calls, the following telephony hardware is recommended: 2 D/240PCI-T1 Adapters 220a, 220b 1 D/640SC Voice Resource Adapter 222 1 DCB/640Conferencing Adapter 224 1 CTBUS to SCBus Connector 226 2 SCBus ribbon cables 228a, 228b

Detailed Description Text (129):
In the preferred embodiment, Nuance 6.2.1 is installed on each machine that will be used in the VA platform server set. The Nuance 6.2.1 installation media contains five separate software packages, which should be installed on each system in the following order: 1. Nuance 6.2.1 Core Software 2. Dialog Optimizer Beta 1) 3. Java Plug-n (Version 1.1.2) 4. Nuance Grammar Builder (Beta 1) 5. Foundation Speech Objects (Beta 1)